SUN BLUEPRINTS
APPROVED

# Repairing File Ownership and Mode

*By Richard Elling - Enterprise Engineering*

*Sun BluePrints™ OnLine - December 1999*

Sun microsystems
ENTERPRISE ENGINEERING

`http://www.sun.com/blueprints`

**Sun Microsystems, Inc.**
901 San Antonio Road
Palo Alto, CA 94303 USA
650 960-1300 fax 650 969-9131

Part No.: 806-4048-10
Revision 01, December 1999

DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

CETTE PUBLICATION EST FOURNIE "EN L'ETAT" ET AUCUNE GARANTIE, EXPRESSE OU IMPLICITE, N'EST ACCORDEE, Y COMPRIS DES GARANTIES CONCERNANT LA VALEUR MARCHANDE, L'APTITUDE DE LA PUBLICATION A REPONDRE A UNE UTILISATION PARTICULIERE, OU LE FAIT QU'ELLE NE SOIT PAS CONTREFAISANTE DE PRODUIT DE TIERS. CE DENI DE GARANTIE NE S'APPLIQUERAIT PAS, DANS LA MESURE OU IL SERAIT TENU JURIDIQUEMENT NUL ET NON AVENU.

Please
Recycle

Adobe PostScript™

# Rapid Recovery Techniques: Repairing File Ownership and Mode

System recovery time, also described as mean time to recover (MTTR), is an important part of availability calculations. Reducing recovery time directly and positively impacts the overall availability of systems.

This is the second article in a series that will discuss rapid recovery techniques for the Solaris™ Operating Environment. The focus of this series is to show ways in which recovery time for repairing or restoring systems can be decreased thus increasing overall availability. This series will also deal with recovery techniques for problems specifically not related to failed hardware. Computer industry analysts observe failures of three main types: product, people, and process. In particular, this series will discuss how to use processes to recover from errors caused by people.

## The Software Registry

The Solaris Operating Environment includes a software registry for maintaining information on software packages and files installed on the system. This registry plays a critical role in software installation, upgrades, and patches. The registry is the primary location for software installation information.

The software registry is a key component of the Solaris Operating Environment because it contains information on what and where software is installed. This information is vital for recovering from errors that involve changes in the software or files on the system. The registry is also invaluable for auditing the system to determine what software has been installed, removed, or patched.

The software registry contains a database of installed files. This database is physically located in the file `/var/sadm/install/contents`, hereafter referred to as the `contents` file. Each file, special file, and directory installed on the system has an entry in this database. Information about each file includes:

- full pathname where the file is installed
- type of file: normal file, editable, volatile, directory, link, raw, cooked, etc.
- a secondary descriptor field, as needed
- file mode (often called permissions) in octal (see `chmod(1)`)
- owner id
- group id
- 16-bit checksum from the `sum` command
- size in bytes
- package name(s) that installed the file

The `pkgchk` command can provide a verbose description of the information in the `contents` file. For example:

```
$ pkgchk -l -p /usr/bin/ls
Pathname: /usr/bin/ls
Type: regular file
Expected mode: 0555
Expected owner: bin
Expected group: bin
Expected file size (bytes): 18120
Expected sum(1) of contents: 53113
Expected last modification: Oct 06 00:43:05 1998
Referenced by the following packages:
    SUNWcsu
Current status: installed
```

# Finding Changes in Installed Files

The information in the software registry can be used to determine changes made to the installed files. By default the pkgchk command will list any discrepancies between the software registry and the system. This is accomplished by examining every entry in the contents file and checking the ownership, mode, size, and checksum. Any discrepancies are reported on stderr. For example:

```
# pkgchk
...
ERROR: /devices/pseudo/cn@0:console
    owner name <root> expected <rme> actual
...
ERROR: /etc/default/init
    file size <219> expected <462> actual
    file cksum <18231> expected <38691> actual
...
ERROR: /etc/dumpdates
    file size <0> expected <420> actual
    file cksum <0> expected <25060> actual
...
ERROR: /etc/inet/hosts
    file size <46> expected <244> actual
    file cksum <3463> expected <16840> actual
...
ERROR: /proc
    permissions <0755> expected <0555> actual
    group name <sys> expected <root> actual
...
ERROR: /etc/rc2.d/S82mkdtab
    pathname does not exist
    pathname not properly linked to </etc/init.d/mkdtab>
...
```

In this example each ERROR has a logical explanation. At present there is not a comprehensive list of these exceptions and they will vary from system to system. For the previous example, these are reconciled as follows:

```
ERROR: /devices/pseudo/cn@0:console
    owner name <root> expected <rme> actual
```

At Solaris Operating Environment install time, the console device is owned by root. Upon login at the console port, the console owner will be changed to the user who logged in.

```
ERROR: /etc/default/init
    file size <219> expected <462> actual
    file cksum <18231> expected <38691> actual
```

The `/etc/default/init` file contains the default system timezone variable, `TZ`. For many systems this will be set to something other than the factory default.

```
ERROR: /etc/dumpdates
    file size <0> expected <420> actual
    file cksum <0> expected <25060> actual
```

At Solaris Operating Environment installation time there would have been no invocations of `ufsdump`. The `/etc/dumpdates` file is updated by `ufsdump` with the ‘`u`’ option. The `/etc/dumpdates` file tracks the backups performed by `ufsdump` over time and is expected to change.

```
ERROR: /etc/inet/hosts
    file size <46> expected <244> actual
    file cksum <3463> expected <16840> actual
```

The `/etc/hosts` file is actually a symbolic link to `/etc/inet/hosts`. This file generally contains the IP addresses of the network interfaces on the system and is expected to differ from the factory default.

```
ERROR: /proc
    permissions <0755> expected <0555> actual
    group name <sys> expected <root> actual
```

The Solaris Operating Environment process file system is mounted at the `/proc` mount point. The `/etc/vfstab` file contains the entry for mounting `/proc` at boot time. However the mount point must exist as a directory prior to the mount. When Solaris Operating Environment is installed, the `/proc` directory is created and has an entry in the software registry. At boot time the process file system will be mounted over the /proc directory. In this case the permissions and group differ between the mount point directory and the mounted file system.

```
ERROR: /etc/rc2.d/S82mkdtab
    pathname does not exist
    pathname not properly linked to </etc/init.d/mkdtab>
```

The /etc/rc2.d/S82mkdtab file is installed as a hard link to /etc/init.d/mkdtab. The S82mkdtab script will run once and then remove itself from the /etc/rc2.d directory. However, the S82mkdtab script does not remove the entry in the software registry database.

# Files Expected to Change

A number of files are expected to change such as /etc/inet/hosts. These files are usually editable or volatile. pkgchk has a -n option that will bypass checking these files. Though this is a tempting option to use for reducing the amount of output from an audit, it is best to see everything that has changed.

```
$ pkgchk -l -p /etc/inet/hosts
Pathname: /etc/inet/hosts
Type: editted file
Expected mode: 0444
Expected owner: root
Expected group: sys
Referenced by the following packages:
        SUNWcsr
Current status: installed
```

# Auditing Solaris™ Software Installations

Although the simple execution of pkgchk without any options will check the installed software on the system, it does not check all installed files. pkgchk by itself will only check those files that are in the software registry database. However, almost all systems have additional files that are not likely to be in the registry. A complete system audit can determine both discrepancies and unregistered files. For example:

```
# find / -mount -exec pkgchk -p {} \;
...
WARNING: no information associated with pathname </etc/default/
dhcp>
...
```

In this example, `pkgchk` prints a WARNING because the file `/etc/default/dhcp` was discovered but is not in the registry. This exception is fine for the system because the `/etc/default/dhcp` file is created by the `dhcpconfig` command.

A comprehensive system software audit should include all directories where software is installed and registered with the Solaris software registry. This typically includes the `/`, `/usr`, `/var`, and `/opt` directories. Other directories such as `/home` will contain files that are not in the registry and, therefore, do not need to be checked against it.

In many Solaris systems the `/`, `/usr`, `/var`, and `/opt` directories are in different file systems. The `-mount` option to the `find` command will restrict `find` to only the initial file system checked. The following example performs a comprehensive system software installation audit:

```
# df
/      (/dev/dsk/c0t0d0s0):  481710 blocks   145079 files
/usr   (/dev/dsk/c0t0d0s5):  684910 blocks   249982 files
/var   (/dev/dsk/c0t0d0s4):   15104 blocks    46950 files
/opt   (/dev/dsk/c0t1d0s0):  360164 blocks   462345 files
...
# find / -mount -exec pkgchk -p {} \;
...
# find /usr -mount -exec pkgchk -p {} \;
...
# find /var -mount -exec pkgchk -p {} \;
...
# find /opt -mount -exec pkgchk -p {} \;
...
```

Using `find` and executing `pkgchk` for each file will consume far more system resources than using `pkgchk` without options as shown in the previous example. While each execution of `pkgchk  -p` consumes 1 to 2 seconds of CPU time, there may be several thousand executions of pkgchk by find. The advantages of using find rather than pkgchk without options are: files not in the registry are detected, and searches can be restricted to specific portions of the directory structure.

# Using Alternate File Databases

By default pkgchk will use the contents file as the installed file database. This is sufficient for most purposes. There are cases where more flexibility is desired.

## Security Audits

pkgchk checks the file owner, group, mode, length in bytes, and checksum. These are compared against the expected values in the contents file. However, an intruder who has compromised the root user account can easily adjust the contents file to accurately reflect any changes made. pkgchk does not have an option for explicitly specifying an alternate contents file. Possible solutions to this problem include: recovering an archived version of the contents file for comparison, and keeping a copy of the contents file available on the system for comparison.

pkgchk has a -m option for specifying a pkgmap file instead of the contents file. The pkgmap format is significantly different from the contents file format. Each package and patch includes a pkgmap file. Many packages, including Solaris software, are distributed on CD-ROM media that resist tampering. However, this becomes cumbersome because each patch must also be reconciled. For example:

```
# cd /cdrom/sol_7_599_sparc_sun_srvr/s0/Solaris_2.7/Product/
SUNWcsr
# pkgchk -m pkgmap
...
ERROR: /sbin/sulogin
    file size <288544> expected <292328> actual
    file cksum <63023> expected <51847> actual
...
```

In this example, the distribution CD-ROM for Solaris 7 5/99 is used to verify the Core Solaris (root), SUNWcsr, package. The file /sbin/sulogin shows an unexpected change in size and checksum. sulogin is an important file that involves system security and should be closely scrutinized. Further investigation reveals that /sbin/sulogin was patched by patch 106541-04. Patches have pkgmap files and patch installation properly updates the `contents` database. But patches cannot change the original package. This shows that using `pkgmap` with the `-m` option is possible but cumbersome for comprehensive security audits.

## Diskless Clients

`pkgchk` has a `-R` *`root_path`* option for checking diskless clients. All files, including the package system information files, are located in a directory tree starting at the specified *`root_path`*. The `-R` option is useful for checking diskless clients from the server. For example:

```
# pkgchk -R /export/root/client1
...
```

The `-R` option is also useful for checking systems from an alternate boot device. For this example, a server with Solaris Operating Environment installed on `/dev/dsk/c0t0d0s0` is booted from a network boot server:

```
{1}ok boot net -s
...
# fsck /dev/rdsk/c0t0d0s0
** /dev/rdsk/c0t0d0s0
** Last Mounted on /
** Phase 1 - Check Blocks and Sizes
** Phase 2 - Check Pathnames
** Phase 3 - Check Connectivity
** Phase 4 - Check Reference Counts
** Phase 5 - Check Cyl groups
36980 files, 738911 used, 223671 free (2247 frags, 27678 blocks,
0.2% fragmentation)
# mount /dev/dsk/c0t0d0s0 /a
# pkgchk -R /a
...
```

# Repairing File Ownership and Mode

pkgchk has a -f option which will correct file attributes if possible. File attributes (owner, group, and mode) will be set to those specified in the contents file or pkgmap file. Contrary to the man page on pkgchk(1M), setuid, setgid, and sticky bits may also be set. Missing directories, named pipes, links, and special devices will be created if they do not already exist.

The pkgchk -f option can be very useful for recovering from unintended ownership or mode (permissions) changes. chown, chgrp, and chmod commands all have a recursive, -R, option. The recursive option will descend the directory structure and change all files and directories found. As such they are prone to accidents. In my experience, these accidents occur far too frequently. In one case, the system administrator accidentally typed "chmod -R 777 /usr". This made the system unusable because the /usr/bin/login program must be setuid root to allow a user to complete the login sequence.

Rebooting, a method popular among many desktop oriented operating system users, does nothing to restore the state of the files on disk. One solution would be to completely reinstall the operating system and any other software that would be installed under the /usr directory. Another solution would be to restore the /usr directory structure from tape backup. These are very time consuming tasks and may not be considered in the class of rapid recovery techniques.

A better solution is to use the Solaris software registry and the pkgchk -f option to restore the modes. If necessary, an alternate boot device can be used and pkgchk with the -R option to repair the problem file system. In any case, a software audit using pkgchk would show the changed files that need attribute repair.

Repairing attributes using the pkgchk -f option should be used only with the -p *path* option. By default, pkgchk will check all files in the contents file. For a reasonably sized server this may include tens of thousands of files. The -f option does not provide feedback when it is actually making corrections nor does it prompt the user whether or not to make corrections. In this regard the -f option is almost as prone to accidental use as the recursive option to chown, chgrp, and chmod. It is best to audit the files using pkgchk to create the list of files that require attribute repairs and then repair with the pkgchk -f and -p *path* options.

## Tips for Using pkgchk

The following tips are for Solaris system administrators who want to use the software registry for configuration management:

- Use `pkgchk` to quickly audit all files in the Solaris software registry.
- Use `find` with `pkgchk` for a more comprehensive audit of installed software.
- Use `pkgchk` with the `-m` option to check installed files against the original package or patch.
- Use `pkgchk` with the `-R` option to check diskless clients or systems booted from alternate boot devices.
- Use `pkgchk` with the `-f` and `-p` *`path`* options to repair file attributes.

## Conclusion

This article describes the Solaris Operating Environment software registry and how to use the `pkgchk` command to audit software installations. This information is important for maintaining the software configuration of a Solaris system. Using `pkgchk` to rapidly recover a system with improper file attributes is also described.

### *Author's Bio: Richard Elling*

***Richard is a Senior Engineer in Enterprise Engineering for the Computer Systems at Sun Microsystems in San Diego, California. Richard had been a field systems engineer at Sun for five years. He was the Sun Worldwide Field Systems Engineer of the Year in 1996. Prior to Sun, he was the Manager of Network Support for the College of Engineering at Auburn University, a design engineer for a startup microelectronics company, and worked for NASA doing electronic design and experiments integration for Space Shuttle missions.***